Contact Tracing over Uncertain Indoor Positioning Data

Tiantian Liu, Huan Li, *Member, IEEE*, Hua Lu, *Senior Member, IEEE*, Muhammad Aamir Cheema, *Senior Member, IEEE*, and Harry Kai-Ho Chan, *Member, IEEE*

Abstract—Pandemics often cause dramatic losses of human lives and impact our societies in many aspects such as public health, tourism, and economy. To contain the spread of an epidemic like COVID-19, efficient and effective contact tracing is important, especially in indoor venues where the risk of infection is higher. In this work, we formulate and study a novel query called Indoor Contact Query (ICQ) over raw, uncertain indoor positioning data that digitalizes people's movements indoors. Given a query object o, e.g., a person confirmed to be a virus carrier, an ICQ analyzes uncertain indoor positioning data to find objects that most likely had close contact with o for a long period of time. To process ICQ, we propose a set of techniques. First, we design an enhanced indoor graph model to organize different types of data necessary for ICQ. Second, for indoor moving objects, we devise methods to determine uncertain regions and to derive positioning samples missing in the raw data. Third, we propose a query processing framework with a close contact determination method, a search algorithm, and the acceleration strategies. We conduct extensive experiments on synthetic and real datasets to evaluate our proposals. The results demonstrate the efficiency and effectiveness of our proposals.

1 Introduction

In the last 20 years, many pandemics broke out all over the world. For example, SARS spread to 20+ countries, sickening over 8,000 people and killing nearly 800 before it vanished in 2003 [1]. Swine flu (H1N1) killed around 600 thousand people worldwide and still circulates in the human population today [1]. As of the end of July 2022, the most recent COVID-19 pandemic has caused more than 561 million infection cases globally, including over 6 million deaths since 2019[1]. These pandemics have caused dramatic losses of human lives and impacted our societies in many aspects such as public health, tourism, and economy.

To contain the spread of such an epidemic, several measures have been proposed and practised, e.g., wearing masks, keeping social distance, and vaccination. Apart from these measures, it is of high importance to conduct efficient (i.e., timely) and effective contact tracing, e.g., finding likely infected people who had close contact with a person whose infection was already confirmed. This is especially relevant in many indoor venues such as railway stations and airports where people travel from and to many different cities, and malls that accommodate many people when restrictions are loosened or lifted. All such indoor venue scenarios call for efficient and effective contact tracing. For example, if one person who visited a mall is confirmed to be infected, finding others in close contact with this person is necessary for timely appropriate measures like testing and quarantine to break the chain of infection. Otherwise, the failure of contact tracing for such an indoor venue would no doubt allow the virus to spread further and infect more people.

In such a case mentioned above, a simple way is to treat all people in the indoor venue at that time as close contact. However, this is very ineffective and brings about a lot of false alarms and unnecessary inconvenience to people. Moreover, it is hard to scale to trace close contacts with many infected people in a building. Thus, we desire to propose an efficient way to analyze the indoor movement of people and identify those that are likely to be close contacts.

In this work, we study a new type of query called Indoor Contact Query (ICQ) over indoor positioning data that digitalizes peoples' indoor movements. Given a query object o, e.g., a person confirmed to be infected with coronavirus, an ICQ analyzes historical indoor positioning data to find those objects (other people) that most likely had close contact with object o for a sufficiently long period of time.

In addition to epidemics, ICQ can have many other uses. For example, in criminal investigations, law enforcement agencies can pinpoint potential suspects by identifying individuals who likely had close contact with the victim. Also, ICQ can be utilized as a tool for identifying associates of a specific individual — if they have had prolonged close contact, it is likely they know each other.

Processing ICQ is challenged by several substantial technical difficulties. First, due to the limitations of indoor positioning [2], the quality of raw indoor positioning data is often lower than that of outdoor GPS data, making it difficult to accurately determine indoor uncertainty regions at sampling times that are not present in the raw positioning data. Second, the indoor environment is complex with special entities such as walls and doors, making it difficult to model and process indoor data. Third, ICQ involves a number of constraints, such as distance and contact duration. Such constraints render the query processing more complex, and it is non-trivial to form an appropriate and easy-to-compute formulation of close contact for ICQ.

- *T. Liu and H. Lu are with the Department of People and Technology, Roskilde University, Denmark. E-mail: {tliu, luhua}@ruc.dk*
- *H. Li is with the College of Computer Science and Technology, Zhejiang University, China. E-mail: lihuan.cs@zju.edu.cn*
- *M. A. Cheema is with the Faculty of Information Technology, Monash University, Australia. E-mail: aamir.cheema@monash.edu*
- *H. K.-H. Chan is with the Information School, University of Sheffield, United Kingdom. E-mail: h.k.chan@sheffield.ac.uk*

[1] https://covid19.who.int

Existing solutions fall short in solving ICQ. First of all, the proximity-based contact tracing solutions [3], [4], [5], [6], [7], [8], [9] find a pair of close contact devices if they perceive each other for a certain period. However, these methods need to install apps on devices, and they do not consider spatial information and thus fail to customize close contact criteria. There also exist location-based contact tracing methods [10], [11], [12], [13], [14], [15], [16], [17] that find spatially close people based on their spatial distances in the historical times. However, these methods mainly focus on outdoor spaces and neglect the indoor topology in our problem setting. Besides, there are studies using indoor positioning data for distance-based analysis [18], [19], [20], [21], but they either ignore the uncertainty of the positioning data [18], or fail to analyze the relationship between two individual moving objects [19], [20], or focus on a different, online early warning problem [21].

To enable efficient and effective ICQ answering, we propose a complete set of data management techniques in this paper. First, we design an enhanced graph model to accommodate heterogeneous indoor data, namely the indoor space topology and distances, indoor moving objects and their raw positioning data, and objects' sampled trajectories. All these types of data are necessary for processing ICQ.

Second, since the query is over uncertain indoor positioning data, we need to analyze the uncertain locations for objects when the data contains no positioning records for them at particular times. To this end, we devise a method to determine the indoor uncertainty regions at timestamps that are "unseen" in the raw positioning data. Also, we introduce a method to derive samples based on two consecutive indoor uncertainty regions with respect to the contextual positioning records before and after a particular sampling time of interest.

Third, we design a framework to process ICQ, which searches for all objects in close contact with the query object o by calling the proposed *constrained* search algorithm C-ICQ. C-ICQ is built on a method that determines the close contact by checking instant contacts during the period of interest. We further develop several strategies to avoid unnecessary computations and speed up querying.

Last but not least, we conduct extensive experiments to evaluate the proposed techniques on synthetic and real datasets. The results show that our sampling based contact tracing is effective at searching for real close contact objects, and C-ICQ is efficient at finding them.

In summary, we make the following major contributions.

- First, we formulate Indoor Contact Query (ICQ) for contact tracing over historical uncertain indoor positioning data.
- Second, to adapt to the complex indoor environment and the low quality indoor positioning data, we build technical foundations of ICQ, including an enhanced indoor graph model, the determination of indoor uncertainty regions for moving objects, and the generation of derived samples from uncertain positioning data.
- Third, we propose an ICQ processing framework, together with a close contact determination method, a search algorithm, and several strategies that accelerate the complex ICQ processing.
- Fourth, we conduct extensive experiments to evaluate our proposals on synthetic and real datasets under various settings.

The rest of the paper is organized as follows. Section 2 presents the data, problem, and solution overview. Section 3 lays the technical foundations of ICQ. Section 4 details ICQ processing. Section 5 reports on the extensive experiments. Section 6 reviews related work. Section 7 concludes the paper and discusses future directions.

2 Preliminaries

Table 1 lists the notations frequently used in this paper.

TABLE 1: Notations

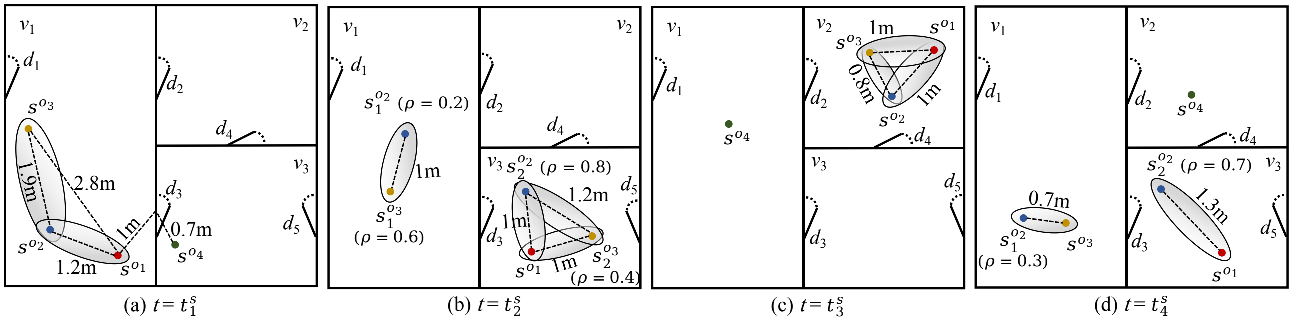
Symbol	Meaning
o, O	indoor object, object set
Ψ_o, Ψ_o^s	o's raw trajectory and sampled trajectory
l	indoor location
$dist(l_i, l_j)$	contact distance between l_i and l_j
t_w^s	a sampling time
$s = (l, \rho)$	a sample at l with probability ρ
$\mathsf{P}(o, o', t_w^s)$	contact probability between o and o' at t_w^s

2.1 Raw and Sampled Indoor Trajectories

In our setting, an indoor positioning system aperiodically reports an indoor moving object o's positioning record as $\psi = (l, t, et)$, which means that object o was found at a location l at time t and the result is expired at a later time et. A location l is captured as (x, y, f), meaning a point (x, y) on a floor f. Each object o's **raw trajectory** Ψ_o is a time-ordered sequence of all its positioning records. In such a raw trajectory Ψ_o, two consecutive positioning records $\psi_i = (l_i, t_i, et_i)$ and $\psi_{i+1} = (l_{i+1}, t_{i+1}, et_{i+1})$ satisfy that $et_i < t_{i+1}$. Due to the discrete nature of indoor positioning, the records of an indoor trajectory Ψ_o most often do not fully disclose the object whereabouts during Ψ_o's lifespan.

To enable the comparison of two objects' trajectories throughout a common part of their lifespans, we generate the **sampled trajectory** of each object as follows. First, we set a unified sampling time interval Δt, and sample the object locations at each sampling time $t_w^s = t_o + w \cdot \Delta t$ where t_o is the global origin time and w is a non-negative integer. Subsequently, the location sample at each sampling time t_w^s is generated in two cases. If t_w^s falls within the time range $[t_i, et_i]$ of a record ψ_i, a sample $s = (l_i, 1)$ is generated, meaning that o is located at l_i with the probability of 1. Such a sample is directly obtained from the raw positioning record and thus is called an **original sample**. Otherwise, the two positioning records before and after the sampling time t_w^s are obtained, and a set S_w of samples in the form of $s_i = (l_i, \rho_i)$ are derived based on the two obtained positioning records. Such a sample is called a **derived sample**. Each sampled location l_i is associated with a probability ρ_i and we have $\sum_{s_i \in S_w} s_i.\rho_i = 1$. How to obtain the location samples based on the contextual positioning records will be detailed in Section 3.3.

As a result, a **sampled trajectory** Ψ_o^s is obtained as the time-ordered sequence of the sample sets at all sampling times, formally $\langle (S_1, t_1^s), \ldots, (S_W, t_W^s) \rangle$. In this way, all



v_1
v_2
v_3
d_1
d_2
d_3
d_4
d_5
s^{o_1}
s^{o_2}
s^{o_3}
s^{o_4}
1.9m
2.8m
1m
0.7m
1.2m
$s_1^{o_2}$ (ρ = 0.2)
$s_1^{o_3}$ (ρ = 0.6)
$s_2^{o_2}$ (ρ = 0.8)
$s_2^{o_3}$ (ρ = 0.4)
0.8m
$s_1^{o_2}$ (ρ = 0.3)
$s_2^{o_2}$ (ρ = 0.7)
1.3m

(a) $t = t_1^s$ (b) $t = t_2^s$ (c) $t = t_3^s$ (d) $t = t_4^s$

Fig. 1: Indoor objects at four consecutive sampling times (the restriction distance δ = 2 m).

objects' location samples are aligned in the time dimension and the comparison of objects is performed only at the sampling times. Note that an object's sampled trajectory is only generated within the lifespan of its raw trajectory.

2.2 Problem Formulation

In real applications, it is interesting to find out those moving objects that had very close contact with a particular object for a sufficiently long period of time, equivalently a sufficient number of consecutive sampling times from the perspective of sampled trajectories. In this paper, we propose a self-contained set of contact tracing techniques that allow us to efficiently find close contacts to a given object by using uncertain indoor positioning data.

We first define the *contact distance* as follows.

Definition 1 (Contact Distance)**.** *The contact distance between two indoor locations l_i and l_j is computed as*

$$dist(l_i, l_j) = \begin{cases} ||l_i, l_j||_E, & \text{if } l_i \text{ and } l_j \text{ fall in the same partition;} \\ \infty, & \text{otherwise.} \end{cases} \quad (1)$$

where $||\cdot,\cdot||_E$ computes the Euclidean distance.

In the definition above, a *partition* refers to a basic topological unit of the indoor space, e.g., rooms, staircases, and elevators. Two persons are not considered contact if they are in different partitions[2]. For example, in Fig. 1 (a), the contact distance between s^{o_2} and s^{o_3} is the Euclidean distance 1.9 meters, while the contact distance between s^{o_2} and s^{o_4} is infinite because they are not in the same partition. Given the distance threshold δ in contact restriction, two objects' *contact probability* at a sampling time t_w^s is defined as follows.

Definition 2 (Contact Probability)**.** *Given two indoor objects o_i and o_j, we obtain their sample sets at time t_w^s as $S_{i,w}$ and $S_{j,w}$. The contact probability is computed as*

$$\mathsf{P}(o_i, o_j, t_w^s) = \sum_{(s_i,\rho_i)\in S_{i,w}, (s_j,\rho_j)\in S_{j,w}} c(l_i, l_j)\rho_i\rho_j, \quad (2)$$

where $c(l_i, l_j)$ is 1 if the contact distance violates the restriction and 0 otherwise.

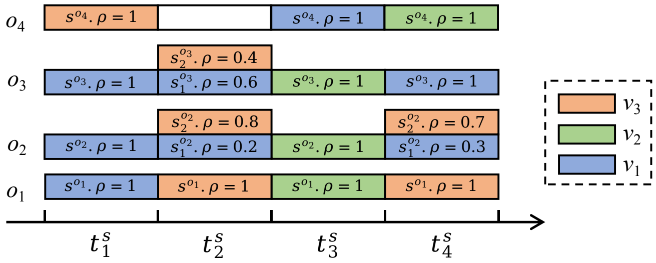


Fig. 2: Temporal view of objects.

We say two objects are **instant contact** with each other at time t if their contact probability is higher than a specified threshold η.

Example 1. *Take a look at Fig. 1 (b) and Fig. 2. We calculate the contact probability $\mathsf{P}(o_2, o_3, t_2^s)$ as follows. At time t_2^s, the object o_2 has two samples, one in the partition v_1 with a probability of 0.2 and the other in v_3 with a probability of 0.8; the object o_3 has a sample in v_1 with a probability of 0.6 and another in v_3 of a probability of 0.4. In both v_1 and v_3, the two samples of o_2 and o_3 are within a distance less than the restriction distance 2 m. As a result, $\mathsf{P}(o_2, o_3, t_2^s) = 0.2 \times 0.6 + 0.8 \times 0.4 = 0.44$.*

Finally, our research problem is formulated as follows.

Problem (Indoor Contact Query, ICQ)**.** *Given a query object o, a time interval T, a distance constraint δ, a contact probability threshold η, and a contact number k, an indoor contact query* ICQ(o, T, δ, η, k) *returns all moving objects having instant contacts with o (i.e., their contact probability is no less than η) for at least k consecutive sampling times within T. In particular, the contact number k helps determine the time period during which two objects are judged to be close contacts.*

Example 2. *Figs. 1 and 2 exemplifies the indoor objects at four consecutive sampling times. Given a query* ICQ$(o_2, [t_1^s, t_4^s], 2, 0.5, 3)$, *the close contact between o_2 and o_i at each sampling time is determined by the threshold $\eta = 0.5$. Referring to o_3, the contact probabilities between o_2 and o_3 at sampling times t_1^s to t_4^s are $[1, 0.44, 1, 0.3]$. As the contact probabilities at t_2^s and t_4^s are less than the threshold 0.5, o_3 is not returned. In contrast, the contact probabilities between o_2 and o_1 are $[1, 0.8, 1, 0.7]$ from t_1^s to t_4^s, which all exceed $\eta = 0.5$. As there are 4 ($> k = 3$) consecutive instant contacts, o_1 is determined as a close contact of o_2 and thus returned. Likewise, o_3 is returned due to its four consecutive instant contacts with o_2.*

[2] In certain situations, objects located near a door but in separate partitions, such as s^{o_1} and s^{o_4} in Fig. 1, may be considered in contact. Our method can be easily adapted to cover this case by linking an object near a door within a certain distance to both adjacent partitions in the proposed enhanced indoor graph model in Section 3.1.

2.3 Solution Overview

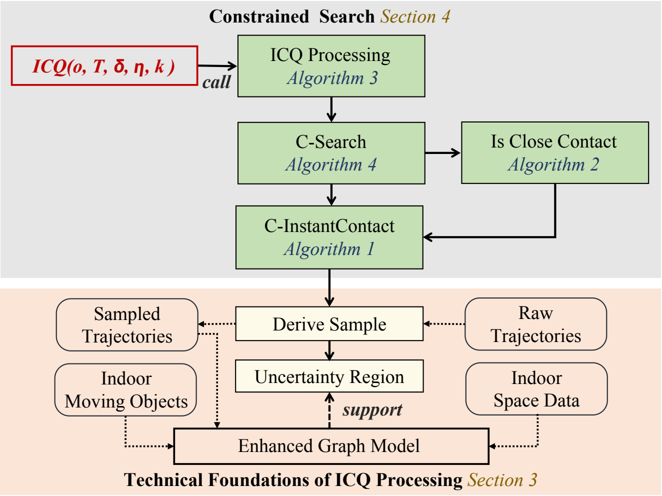


Fig. 3: The solution for indoor contact query.

Fig. 3 shows our solution for ICQ. The bottom layer lays the technical foundations of ICQ processing, as to be detailed in Section 3. In particular, an enhanced graph model is constructed to accommodate the indoor space data, indoor moving objects, and sampled trajectories for relevant computations. On top of the enhanced graph model, it first finds a moving object's uncertainty region, i.e., the object's possible locations at a sampling time. Based on the uncertainty regions, it derives samples which in turn are maintained in the enhanced graph model for subsequent use.

To process a query instance ICQ(o, T, δ, η, k), an overall framework is proposed as Algorithm 3. It finds all objects in close contact with the query object o by calling Algorithm 4 (the constrained search), which employs a set of strategies for acceleration. Then, Algorithm 4 calls Algorithm 1 to determine the instant close contact and further help set the start timestamp. It also calls Algorithm 2 for close contact determination when processing a candidate object. Algorithm 2 calls Algorithm 1 to compute the contact probability at an instant timestamp. We present the close contact determination in Section 4.1 and the ICQ search in Section 4.2.

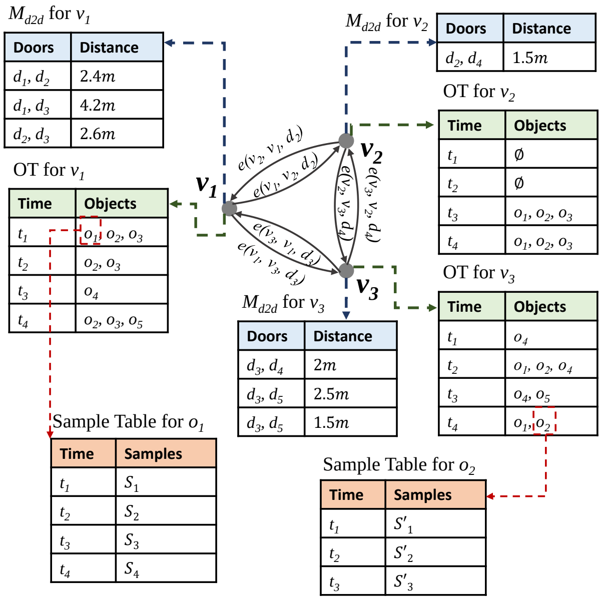
It is noteworthy that Algorithms 1 calls a *Derive Sample* function when computing contact probabilities over object samples, whose details can be found in an extended version (https://arxiv.org/abs/2304.08838).

3 Foundations of Query Processing

This section lays the technical foundations of ICQ processing. Section 3.1 introduces an enhanced graph model to organize indoor data for relevant computations. Section 3.2 determines the indoor uncertainty regions at sampling times that are "unseen" in its raw positioning records. Section 3.3 describes the generation of derived samples based on the two indoor uncertainty regions with respect to the contextual positioning records before and after. Section 3.4 introduces the indoor data preprocessing.

3.1 Enhanced Graph Model

We design an enhanced graph model to organize three kinds of data, namely the indoor space information, indoor moving objects, and their sampled trajectories. The enhanced graph model is depicted in Fig. 4.



M_{d2d} for v_1

Doors	Distance
d_1, d_2	$2.4m$
d_1, d_3	$4.2m$
d_2, d_3	$2.6m$

OT for v_1

Time	Objects
t_1	o_1, o_2, o_3
t_2	o_2, o_3
t_3	o_4
t_4	o_2, o_3, o_5

Sample Table for o_1

Time	Samples
t_1	S_1
t_2	S_2
t_3	S_3
t_4	S_4

M_{d2d} for v_2

Doors	Distance
d_2, d_4	$1.5m$

OT for v_2

Time	Objects
t_1	∅
t_2	∅
t_3	o_1, o_2, o_3
t_4	o_1, o_2, o_3

M_{d2d} for v_3

Doors	Distance
d_3, d_4	$2m$
d_3, d_5	$2.5m$
d_3, d_5	$1.5m$

OT for v_3

Time	Objects
t_1	o_4
t_2	o_1, o_2, o_4
t_3	o_4, o_5
t_4	o_1, o_2

Sample Table for o_2

Time	Samples
t_1	S'_1
t_2	S'_2
t_3	S'_3

Fig. 4: The example of enhanced indoor graph model.

Following a previous work [22], the core part of the model captures the indoor space topology and distances by a directed and labeled graph $G(V, E, L_V)$ where

1) V is the set of vertices, each for an indoor partition.[3]
2) E is the set of directed edges. An edge $e(v_i, v_j, d_k) \in E$ means one can enter a partition v_j from a partition v_i via a door d_k.
3) L_V is the set of vertex labels. Each label is a three-tuple $(v_i, \uparrow M_{d2d}, \uparrow OT)$ where v_i refers to the associated vertex, $\uparrow M_{d2d}$ is a pointer to the matrix M_{d2d} that stores the intra-partition distance between each pair of doors within v_i, and $\uparrow OT$ is a pointer to the object table that maintains the bucket of objects at each sampling time.

For each partition v_i's object table OT^{v_i}, $OT^{v_i}[t^s]$ maintains all objects that have at least a sample inside v_i at time t^s. Each object o is linked to its corresponding sample table that maintains its set of samples at each sampling time.

In addition to the enhanced graph model, all raw trajectories are stored in a hashtable with object IDs as the keys. Moreover, raw trajectories are indexed by a B-tree based on the positioning records' time attributes. This facilitates obtaining an object's contextual positioning records for a sampling time.

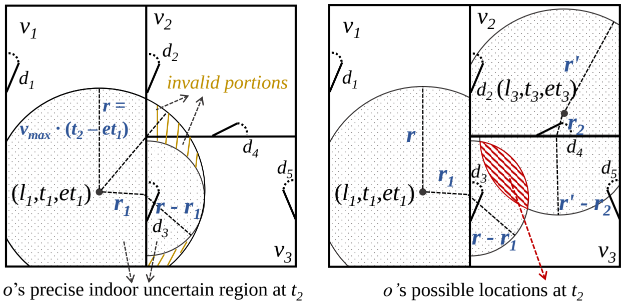
3.2 Indoor Uncertainty Region Determination

Given an object o and a sampling timestamp t^s unseen in the raw trajectory, we obtain a positioning record $\psi = (l, t, et)$ prior to t^s. In a free space, o's possible location at time t^s is constrained by a circle $\bigcirc(l, r)$ centered at l with radius $r = v_{max} \cdot (t^s - et)$, where v_{max} is the maximum object speed.

[3] Indoor partitions can be extracted from indoor geometric data stored in digital files [23], [24].

However, such a circular uncertainty region is imprecise in the presence of indoor topology.

Referring to the example in Fig. 5(a), o's possible location at time t_2 cannot be at the portion of $\bigcirc(l_1, r)$ inside the indoor partition v_2 as o cannot go through the wall to reach v_2 within the distance $r = v_{max} \cdot (t_2 - et_1)$. Indeed, o's possible location at time t_2 is precisely constrained by an *indoor uncertainty region* [19] $UR_I(l_1, r)$ that consists of all indoor portions within the indoor distance r from l_1. As illustrated in Fig. 5(a), $UR_I(l_1, r)$ consists of the shaded part in partition v_1 and the shaded sector centered at the door d_3 with radius $r - r_1$ in partition v_3, where r_1 is the distance from l_1 to d_3. Such indoor portions can be found via an indoor range query, which is formalized in FINDIUR in the extended version.



(a) Precise UR_I

(b) Intersection of Two UR_Is

Fig. 5: The example of the object o's uncertainty regions.

3.3 Generation of Derived Samples

Given an object o's raw positioning record $\psi_\dashv = (l_\dashv, t_\dashv, et_\dashv)$ prior to the unseen sampling timestamp t^s, o's possible location at t^s is constrained by $UR_I(l_\dashv, v_{max} \cdot (t^s - et_\dashv))$. Likewise, o's possible location at t^s should also be constrained by $UR_I(l_\vdash, v_{max} \cdot (t_\vdash - t^s))$ with respect to o's positioning record $\psi_\vdash = (l_\vdash, t_\vdash, et_\vdash)$ immediately after t^s. Taking into account the two contextual positioning records $\psi_\dashv$ and $\psi_\vdash$, o's possible location can only be within the intersection of $UR_I(l_\dashv, v_{max} \cdot (t^s - et_\dashv))$ and $UR_I(l_\vdash, v_{max} \cdot (t_\vdash - t^s))$.

Referring to Fig. 5(b), the red shaded region depicts the intersection of the two indoor uncertainty regions with respect to the contextual records reported at t_1 and t_3. Generally, the intersection of two indoor uncertainty regions is an arbitrary geometry that is difficult to compute. Therefore, we adopt a sample-based approach to approximate the intersection with a set of lattice points contained by the intersection. The approach is formalized in DERIVE in the extended version.

3.4 Data Preprocessing

We do not allow for k' or more consecutive derived sample sets in a trajectory, because that would mean an object absent throughout the k' or more sampling times may still be considered as close contact due to the derived sample sets.

Therefore, we preprocess each object's raw trajectory as follows. We go through the sampling times and check if each of them has a corresponding original sample in the raw trajectory. If no original sample is seen for k' consecutive sampling times, the raw trajectory is split immediately after the raw positioning record that provides the latest original sample. When another original sample is seen at a subsequent sampling time, a new raw trajectory is started from the corresponding raw positioning record. In this way, a raw trajectory may be split into multiple raw trajectories, over each of which at most $k' - 1$ consecutive derived sample sets are generated.

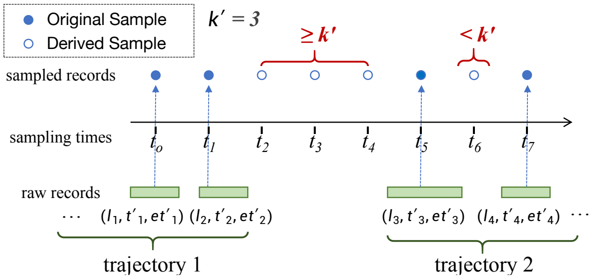


Fig. 6: An example of splitting a raw trajectory ($k' = 3$).

Example 3. *Referring to Fig. 6, the original samples are not seen from the raw positioning records at sampling times t_2 to t_4. Suppose $k' = 3$. Then the current raw trajectory is split after the record (l_2, t'_2, et'_2) that provides the latest original sample at the sampling time t_1. Subsequently, a new trajectory is started from the record (l_3, t'_3, et'_3). This new trajectory will not be split between (l_3, t'_3, et'_3) and (l_4, t'_4, et'_4) because only one derived sample set is needed.*

Our query is processed over the preprocessed trajectories. The preprocessing brings up an important property that facilitates query processing. That is, an object must have an original sample within k' consecutive sampling times. Based on this, we can prioritize the instant contact determination with an original sample of the candidate object while avoiding the step-by-step generation of derived sample sets. This is implemented in line 9 of Algorithm 4, to be detailed in Section 4.2.

4 QUERY PROCESSING ALGORITHMS

This section presents the ICQ query processing algorithms. Section 4.1 introduces the method of identifying all close contacts with the query object o. Section 4.2 details the ICQ processing utilizing a constrained search method. Section 4.3 analyzes the time and space complexity of the proposed solution.

4.1 Close Contact Determination

To find all close contacts with the given object o, we must determine if an object o_i, in the same partition with o, has close contact with o for k consecutive times. To achieve this, we must first devise a method to determine whether two objects are in instant contact. A simple approach would be to sequentially traverse all pairs of samples of these two objects and compute the instant contact probability. If the probability is above the threshold, the two objects are considered instant contact. The method for sequential instant contact determination is described in Algorithm C in the extended version. However, this method is inefficient in time and space (see our analysis in Section 4.3). In the following, we propose three strategies to speed-up the determination.

Strategy 1 (Reverse Early Termination)**.** *At the level of indoor partitions, an object's sample set S is merged as a new set, represented as $\{(v, \sum_{\pi_l(s_i) \in v} \rho_i)\}$, each tuple in which is called a merged sample. Suppose v and v' are two different partitions. For object o's each merged sample (v, ρ_v) and object o''s each merged sample $(v', \rho_{v'})$, the corresponding non-contact probability is computed as $\rho_v \rho_{v'}$. The overall non-contact probability between o and o' is accumulated over all the pairs of different partitions. Once the accumulated non-contact probability reaches or exceeds $1 - \eta$, the contact probability must be lower than η and the two corresponding objects are definitely not in close contact.*

Strategy 2 (Object Sample Pruning)**.** *A pair of two object samples in different partitions is directly pruned as their distance is ∞ (see Equation 1) and larger than the distance constraint δ.*

Strategy 3 (Early Termination)**.** *For two objects and their corresponding sample sets, their contact probability is accumulated by iteratively processing each pair of the two objects' samples. Given the threshold η, two objects are definitely in instant contact once their current accumulated probability reaches or exceeds η.*

With these accelerate strategies, we formalize the constrained instant contact determination in Algorithm 1. First, it uses a caching mechanism such that two objects' instant contact result is directly fetched if the result has been computed and cached (line 1). Then, it prepares the candidate object o''s sample set S' by either calling DERIVE or retrieving from the enhanced graph model (lines 2–3). Next, it merges the samples for o and o', respectively, in lines 4–5. Lines 6–12 implement Strategy 1, in which the overall non-contact probability NP is accumulated and *false* is returned directly if NP exceeds $1 - \eta$. Lines 14–16 implement Strategy 2 such that only a pair of object samples in the common partition v^* is considered in computing concrete instant contact probability. Lines 19–20 implement Strategy 3. After processing each object sample pair, the accumulated contact probability is checked for a potential quick determination. Note that the result is cached before it is returned (see lines 12, 20, and 22).

Then, we can further check whether o' is a close contact of o. ISCLOSECONTACT is formalized in Algorithm 2. It sequentially scans the sampling times from t^s to $t^s + (k-1)\Delta t$. At each scanned time t^q (initilized in line 1), the algorithm calls algorithm C-INSTANTCONTACT to determine if the query object o and the candidate object o' are instant contact. If the two objects have no instant contact at a scanned time t^q, *false* is returned (lines 3–4). Otherwise, *true* is returned to indicate that the two objects are in close contact at all scanned times (line 6).

4.2 ICQ Processing

We proceed to present the overall framework for ICQ processing. Given a query ICQ(o, T, δ, η, k), we search for each possible timestamp and find out the objects that are in contact with the query object o. The main steps are as follows. First, we determine the period $[t^s_s, t^s_e]$ when o appears in the space. Second, we obtain all objects present during $[t^s_s, t^s_e]$. Third, for each such object, we check if it had close contact with o for k consecutive times by 1) generating samples at unseen timestamps and then 2) calculating the instant contact probability at each sampled timestamp to determine the instant contact. Finally, we return all objects that had close contact with o. The overall process is formalized in Algorithm 3.

Algorithm 1 C-INSTANTCONTACT (query object o, candidate object o', current time t^s, distance constraint δ, contact probability threshold η)

```
1:  return the cached result if it exists
2:  if $t^s$ is not seen in $\Psi^s_{o'}$ then $S' \leftarrow$ DERIVE$(o', t^s)$
3:  else obtain $(S', t^s)$ from $\Psi^s_{o'}$
4:  merge samples in $S$ as $\{(v, \sum_{\pi_l(s_i) \in v} \rho_i)\}$
5:  merge samples in $S'$ as $\{(v', \sum_{\pi_l(s'_i) \in v'} \rho'_i)\}$
6:  NP $\leftarrow 0$
7:  for each merged sample $(v, \rho_v)$ do
8:      for each merged sample $(v', \rho_{v'})$ having $v \neq v'$ do
9:          NP $\leftarrow$ NP $+ \rho_v \cdot \rho_{v'}$
10:         if NP $> (1 - \eta)$ then                      ▷ Strategy 1
11:             update the latest non-contact time of $o'$ as $t^s$
12:             cache the result; return false
13: P $\leftarrow 0$
14: for each partition $v^*$ of $S$ and $S'$ do            ▷ Strategy 2
15:     for each sample $(l, \rho)$ in $S$ falling in $v^*$ do
16:         for each sample $(l', \rho')$ in $S'$ falling in $v^*$ do
17:             if $dist(l, l') < \delta$ then
18:                 P $\leftarrow$ P $+ \rho \cdot \rho_i$
19:                 if P $\geq \eta$ then                   ▷ Strategy 3
20:                     cache the result; return true
21: update the latest non-contact time of $o'$ as $t^s$
22: cache the result; return false
```

Algorithm 2 ISCLOSECONTACT (query object o, candidate object o', current time t^s, distance constraint δ, contact probability threshold η, integer k)

```
1: $t^q \leftarrow t^s$
2: while $t^q \leq t^s + (k-1)\Delta t$ do
3:     if !C-INSTANTCONTACT$(o, o', t^q, \delta, \eta)$ then
4:         return false
5:     $t^q \leftarrow t^q + \Delta t$
6: return true
```

Initially, the earliest and latest sampling times to process are obtained (lines 1–2). In particular, the earliest sampling time t^s_s is the older timestamp of the start of the query time interval T and the start of object o's lifespan. The latest sampling time t^s_e is obtained similarly.

Afterwards, the framework prepares the sampled trajectory $\Psi^s_{o'}$ for each object o' that appeared within $[t^s_s, t^s_e]$ (lines 4–12). Specifically, if a sampling time t^s is seen in o''s raw trajectory, the reported location l_i and a probability of 1 form the unique sample at t^s (lines 6–8). Otherwise, the Derive Sample function DERIVE(o, t^s) is called to get the samples at t^s. To avoid heavy computations, here we only derive samples for the query object o since its samples will certainly be used in search (lines 10–11). Other objects' sampling will be conducted during the concrete search. All generated samples will be maintained in the enhanced graph model (see Section 3.1) for subsequent retrieval. Finally, the algorithm calls an efficient search method to find those objects that are in close contact with o during $[t^s_s, t^s_e]$

Algorithm 3 ICQ-PROCESSING (query object o, time period T, distance constraint δ, contact probability threshold η, contact number k)

```
1:  $t^s_s \leftarrow \max(\min(T), \min(\{\psi.t \mid \psi \in \Psi_o\}))$
2:  $t^s_e \leftarrow \min(\max(T), \max(\{\psi.et \mid \psi \in \Psi_o\}))$
3:  $t^s \leftarrow t^s_s$
4:  while $t^s \leq t^s_e$ do
5:      for each object $o'$ that appeared within $[t^s_s, t^s_e]$ do
6:          if $t^s$ is seen in $\Psi_{o'}$ then
7:              obtain record $(l_i, t_i, et_i) \in \Psi_{o'}$, $t^s \in [t_i, et_i]$
8:              add $(\{(l_i, 1)\}, t^s)$ to $\Psi^s_{o'}$
9:          else
10:             if $o'$ is query object $o$ then
11:                 $S \leftarrow$ DERIVE$(o', t^s)$; add $(S, t^s)$ to $\Psi^s_{o'}$
12:     $t^s \leftarrow t^s + \Delta t$
13: return C-SEARCH($o$, $[t^s_s, t^s_e]$, $\delta$, $\eta$, $k$)
```

(line 13).[4] The search method employs constrained search, which avoids unnecessary computations of concrete instant contact probabilities using Strategy 4.

Strategy 4 (Time Skipping)**.** *Given a sampling time t^s, if the candidate object does not have instant contact with the query object, all close sampling times within $[t^s - (k-1)\Delta t, t^s + (k-1)\Delta t]$ are skipped for the candidate object.*

The constrained search (Algorithm 4) works as follows. First, the result set $result$ is initialized (line 1). Then, it initialize each object's latest non-contact time (line 2). Subsequently, the algorithm processes each sampling time t^s sequentially (lines 3–23). For each current time t^s, the sample set S is retrieved from the enhanced graph model and a set $O_{visited}$ is used to record all visited objects in this iteration (line 4). For each sample $s(l, \rho)$ in S, the host partition v is obtained (lines 5–6). The partition v will be skipped (line 7) if it has been visited (see line 8). Next, the algorithm checks each candidate object o_i that has samples[5] in v at t^s (line 9). If o_i is the query object o, or it has been processed, it will be skipped (line 10). Otherwise, o_i is added to the visited set (line 11). The algorithm obtains o_i's last sampling time t^s_{ln} when o_i has no instant contact with o (line 12). Such a non-contact time is initialized as $-\infty$ for all objects (line 2) and updated when determining the instant contact (see lines 11 and 21 in Algorithm 1). For the candidate object o_i that currently has sample(s) in the same partition as o (see line 9), its instant contact with o is determined by calling C-INSTANTCONTACT. After the determination, two cases are differentiated (lines 13–15). If o_i contacts with o at t^q, the end check time t^s_{ec} is updated as t^q, meaning that the sequential checking of close contact should be performed until t^s_{ec}. Otherwise, t^s_{ec} is set to $t^s - \Delta t$ since there is no need to check until the non-contact time t^s. Accordingly, the starting check time t^s_{sc} is also determined (line 16). In particular, t^s_{sc} must be the latest time among the start search time t^s, the sampling time before t^s_{ec} for $(k-1)\Delta t$, and the sampling time next to the last non-contact time t^s_{ln}. The sequential scanning from t^s_{sc} to t^s_{ec} (lines 18–22) is performed only if their time difference exceeds $(k-1)\Delta t$ (line 17). Once the current time t^s is processed, the algorithm moves to the next sampling time $t^s + \Delta t$ (line 23). It returns the result when all sampling times have been processed (line 24).

Algorithm 4 C-SEARCH (query object o, time period $[t^s_s, t^s_e]$, distance constraint δ, contact probability threshold η, contact number k)

```
1:  $result \leftarrow \varnothing$
2:  initialize each object's latest non-contact time as $-\infty$
3:  while $t^s \leq t^s_e$ do
4:      obtain $(S, t^s)$ from $\Psi^s_o$; $O_{visited} \leftarrow \varnothing$
5:      for each $s(l, \rho) \in S$ do
6:          $v \leftarrow host(l)$
7:          if $v$ has been visited then continue
8:          marked $v$ as visited
9:          for each $o_i \in OT^v[t^s]$ do
10:             if $o_i = o$ or $o_i \in O_{visited}$ then continue
11:             add $o_i$ to $O_{visited}$
12:             $t^s_{ln} \leftarrow o_i$'s latest non-contact time
13:             if C-INSTANTCONTACT$(o, o_i, t^s, \delta, \eta)$ then
14:                 $t^s_{ec} \leftarrow t^s$
15:             else $t^s_{ec} \leftarrow t^s - \Delta t$
16:             $t^s_{sc} \leftarrow \max(t^s_s, t^s_{ec} - (k-1)\Delta t, t^s_{ln} + \Delta t)$
17:             if $t^s_{ec} - t^s_{sc} > (k-1)\Delta t$ then        ▷ Strategy 4
18:                 $t^q \leftarrow t^s_{sc}$
19:                 while $t^q + (k-1)\Delta t \leq t^s_{ec}$ do
20:                     if ISCLOSECONTACT($o$, $o_i$, $t^q$, $\delta$, $\eta$, $k$) then
21:                         add $o_i$ to $result$; break
22:                     $t^q \leftarrow t^q + \Delta t$
23:     $t^s \leftarrow t^s + \Delta t$
24: return $result$
```

4.3 Complexity Analysis

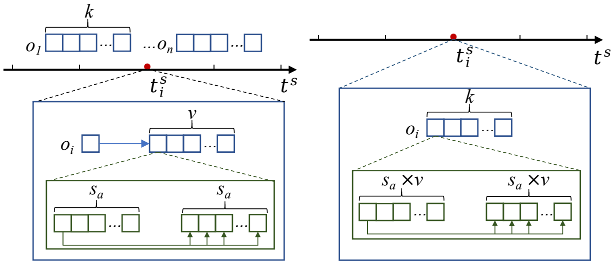
Let $\mathtt{o}$ be the average number of objects falling into the partition of the query object at a time point, and $\mathtt{v}$ be the average number of partitions that an object's samples may fall into at a time point. Generally speaking, we have $\mathtt{o} \ll |O|$ where $|O|$ is the total number of objects, and $\mathtt{v} \ll |V|$ where $|V|$ is the total number of partitions. Let $\mathtt{s}_a$ be the average size of an object' sample sets in a partition at a time point. Let $\mathtt{s}_m$ be the maximal size among all objects' sample sets.

Time Complexity. Fig. 7 illustrates the complexity analysis of (a) the constrained search, and for comparison, (b) the naive sequential search method without using the accelerate strategies.

The proposed constrained search (see Fig. 7(a)) uses a k-sized buffer to maintain the previous contact probabilities computed for each candidate object. Therefore, at each sampling time, the constrained search only processes the current sampling time. Let $\mathtt{o}_p$ be the number of processed candidate objects, where $\mathtt{o}_p \ll \mathtt{o}$ as benefited by Strategy 4 that prunes the candidate objects maintained in the buffer if their probabilities at one time point are less than the threshold η. Note

[4] Within Algorithm 3, the efficient search method could be replaced by a naive sequential search method that determines the instance contacts at each sampling time one by one. It is time-consuming and so we give its details in the extended version.

[5] According to the preprocessing described in Section 3.1, a candidate object must have a certain sample generated by Algorithm 3 within $(k-1)\Delta t$. Therefore, the object will certainly be processed if it is in close contact to o for k consecutive sampling times.



(a) Constrained Search (b) Sequential Search

Fig. 7: Complexity analysis.

that a smaller threshold η and a smaller k will both lower $\mathtt{o}_p$. Given a query object and a candidate object, only $\mathcal{O}(\mathtt{v} \cdot \mathtt{s}_a^2)$ sample pairs are involved in the distances and probabilities calculation as enhanced by Strategy 2 that processes pairs in a partition-by-partition manner. Furthermore, the actual number of distance computations between the sample pairs, denoted by β, is much smaller than $\mathcal{O}(\mathtt{v} \cdot \mathtt{s}_a^2)$ as reduced by Strategies 1 and 3 that enable early termination. As there are in total $(|T|/\Delta t)$ sampling times to process, where $|T|$ is the query interval length and Δt is the unit sampling time interval, the overall time complexity of constrained search is $\mathcal{O}\big((|T|/\Delta t) \cdot \mathtt{o}_p \cdot \beta\big)$.

For comparison, we also include the time complexity analysis of sequential search method as follows. The sequential search (see Fig. 7(b)) calculates the contact probability for each candidate object whose samples appear in the same partition as the query object at each sampling time. For each such object, k consecutive timestamps starting from the current sampling time are processed, each of which involves the distance computation for $(\mathtt{v} \cdot \mathtt{s}_a)^2$ sample pairs. As there are in total $|T|/\Delta t$ sampling times to process, the overall time complexity of the sequential search method is $\mathcal{O}\big((|T|/\Delta t) \cdot k \cdot \mathtt{o} \cdot \mathtt{v}^2 \cdot \mathtt{s}_a^2\big)$. Therefore, the constrained search is at least $\mathcal{O}((\mathtt{o}/\mathtt{o}_p) \cdot k \cdot \mathtt{v})$ times faster than the sequential search, where $\mathtt{o} \gg \mathtt{o}_p$.

In the worst case that all samples of an object are located in one partition, $(\mathtt{v} \cdot \mathtt{s}_a)$ degenerates to $\mathtt{s}_m$, $\mathtt{o}$ and $\mathtt{o}_p$ degenerate to $|O|$, and β degenerates to $\mathtt{s}_m^2$. In this case, the constrained search has the time complexity of $\mathcal{O}\big((|T|/\Delta t) \cdot |O| \cdot \mathtt{s}_m^2\big)$, while the sequential search method is in $\mathcal{O}\big((|T|/\Delta t) \cdot k \cdot |O| \cdot \mathtt{s}_m^2\big)$.

Space complexity. Overall, the space complexity analysis of the constrained search follows the same spirit of its time complexity, but some extra space overheads are needed. In particular, it uses a set to maintain all $|O|$ objects, and also maintains a k-sized buffer for the $\mathtt{o}'$ candidate objects at each sampling time, where $\mathtt{o} \leq \mathtt{o}' \leq |O|$. Besides, at each sampling time, the allocated space will be collected. Therefore, the space complexity is invariant to the number of sampling times, i.e., $(|T|/\Delta t)$. To sum up, the space complexity of the constrained search is $\mathcal{O}\big(\mathtt{o}_p \cdot \beta + |O| + k \cdot \mathtt{o}'\big)$ on average and $\mathcal{O}\big(|O| \cdot (\mathtt{s}_m^2 + k + 1)\big)$ in the worst case.

5 Experiments

5.1 Overall Experimental Settings

All algorithms are implemented in Java and run on a PC with an Intel Core i5 3.10 GHz CPU and 8 GB memory.

Baseline Methods. As no method exists to solve indoor contact query, we design three baseline methods to compare with our proposed search method C-ICQ. First, an S-ICQ method sequentially processes each sampling time and computes the concrete contact probabilities to decide if a candidate object has close contact with the query object[6]. Second, an E-ICQ method finds the uncertainty region at each unseen sampling timestamp based on Euclidean distance rather than indoor distance. Third, an R-ICQ method searches for close contact objects over raw trajectories without uncertainty analyses at unseen timestamps. An object is considered as a close contact if its consecutively observed instant contacts with the query object cover the required contact time. All baselines employ the indoor distance-aware model [22] and a trajectory table for data organization.

Parameters. The settings of query inputs $|T|$, δ, η, and k are listed in Table 2, where the default values are in bold. In practice, they can often be set to default values and users may only change them when necessary. We also vary the object number $|O|$ and the side length ll of lattices (see Section 3.3) which is determined by the system developers and is usually stable values in real-world applications. The unified sampling time interval Δt is set to 10 seconds. Therefore, we vary the instant contact number k as integer multiples of 6 (i.e., 6, 12 and so on) such that the required contact period will be 1, 2 or more minutes.

TABLE 2: Parameter Settings

Parameter	Meaning	Setting
$\|T\|$ (hour)	query interval length	6, 12, 18, **24**
δ (m)	distance constraint	1, **2**, 3, 4, 5
η	probability threshold	0.3, 0.4, **0.5**, 0.6, 0.7
k	instant contact number	6, 12, **18**, 24, 30, 36, 42
ll (m)	lattice side length	0.2, **0.4**, 0.6, 0.8, 1.0
$\|O\|$	object number	2k, **4k**, 6k, 8k

Performance Metrics. To evaluate the efficiency of query processing, we run each query instance 30 times and measure the *average running time* and *average memory cost*.

As there are unseen timestamps in raw trajectories and we have to analyze the uncertain region of an object at an unseen timestamp, it will introduce inaccuracies. Therefore, we evaluate the effectiveness of the query result as follows. We consider the metric *recall*, the fraction of ground truth close contact objects that are included in a query result. Moreover, we use the comprehensive metric *F1 score* that is computed as $2 * (recall \times precision)/(recall + precision)$, where $precision$ measures the fraction of returned close contact objects that are in the ground truth[7]. The closer the F1 score is to 1, the better is the overall effectiveness of the query result.

5.2 Experiments on Synthetic Data

5.2.1 Settings

Indoor Space. We generate a 5-floor indoor space based on a real-world floorplan[8] that is of 1,368 × 1,368 square metres

[6] S-ICQ is implemented using the overall framework in Algorithm 3 whose line 13 instead calls Algorithm C in the extended version.

[7] We have also evaluated $precision$. However, the $precision$ measures in most tests are close to 1. Therefore, we omit them in the evaluation results.

[8] https://longaspire.github.io/s/fp.html

(m^2) with 141 partitions and 216 doors on each floor. Four staircases, each set to 20 m long, connect each two adjacent floors.

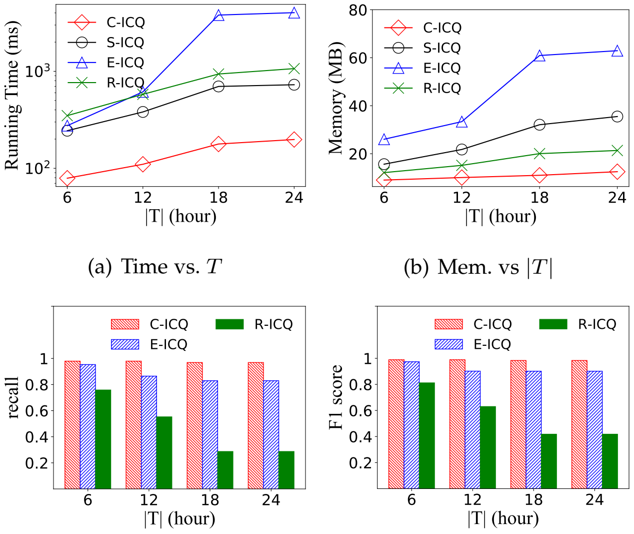
Trajectory Data. We simulate raw trajectories of moving objects in the indoor space for 24 hours (i.e., 86,400 seconds). Each trajectory's lifespan is at least one hour (3,600 seconds) and its start time is random within the first 82,800 seconds. For each object, we randomly select a list of destinations. Between each two consecutive destinations, the object moves along a planned path with fluctuations in direction and moving speed. The maximum speed is set to 5 km/h (approximately 1.4 m/s). At each destination, the object stays for a random period of time from 0 to 480 seconds.

Raw positioning records are generated for each object every 10 seconds. The expiration $et - t$ in each record is fixed to 5 seconds. To simulate the uncertainty in raw trajectories, we randomly remove 10% of the generated records from each original trajectory. After that, we use the preprocessing method described in Section 3.1 to split the resultant trajectories.

Query Instances. For each specific parameter combination, 20 query instances are generated with random query objects and their trajectories. Due to the randomness of trajectory data, it is hard to find close contacts that last for a long period. To increase the likelihood of non-empty query results, we intentionally simulate 10 "close contact trajectories" for each query instance. Each such trajectory is synthesized as follows. From the current query instance's trajectory, we extract a segment that lasts for a random period of 500 to 1800 seconds. The extracted segment is used as the basis of a close contact trajectory. To reduce the trajectory's data quality (and to increase the difficulty for contact tracing), we randomly delete 10% of its positioning records and add some location offsets to some of its positioning records.

5.2.2 Results

Effect of $|T|$. We vary the query interval length $|T|$ from 6 to 24 hours. The running time and memory cost of the four methods are reported in Figs. 8(a) and 8(b), respectively. A longer $|T|$ leads to more time and memory costs as more candidate objects and sampling times are involved. For both efficiency measures, C-ICQ performs the best. This is mainly due to its use of time skipping (Strategy 4) for reducing the calls of instant contact determination. Compared to S-ICQ, C-ICQ additionally introduces early termination (Strategies 3 and 1) and object sample pruning (Strategy 2) to instant contact determination, thereby incurring less time and memory overhead. R-ICQ and E-ICQ cost more time and memory compared to our proposed methods. First, their used data structures (indoor distance-aware model plus a trajectory table) are less efficient than our enhanced graph model that is specialized to the retrieval of samples for relevant objects. Second, for E-ICQ, Euclidean distance leads to a larger uncertainty region with more location samples, which costs more time and memory in contact distance computations. Third, unlike other methods, R-ICQ needs to find and verify all candidate objects that have instant contact with the query object, thus incurring a longer running time. Interestingly, R-ICQ does not involve location samples and thus consumes relatively low memory. Still, C-ICQ consumes even less memory because of its efficient time skipping strategy.



(a) Time vs. T

(b) Mem. vs $|T|$

(c) Recall vs $|T|$

(d) F1 score vs $|T|$

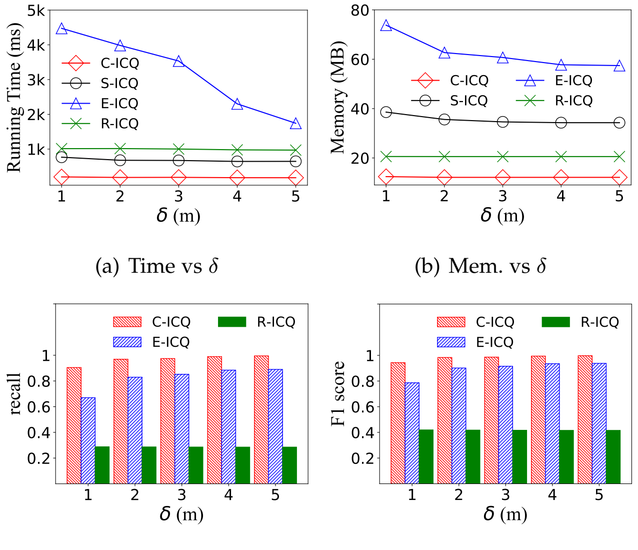
Fig. 8: Effect of $|T|$

The recall and F1 scores are reported in Figs. 8(c) and 8(d), respectively. As S-ICQ and C-ICQ return the same set of close contact objects, we omit S-ICQ in all effectiveness studies. When increasing $|T|$, more candidate objects have to be analyzed. As a result, all methods' two effectiveness measures degrade. Nevertheless, C-ICQ outperforms E-ICQ and R-ICQ at all $|T|$ values. R-ICQ does not analyze the uncertainty region, and its two effectiveness measures are clearly lower that the other methods using a sampling based analysis. This indicates that our sample-based contact distance is effective at finding real close contacts. Although E-ICQ also uses the sampling based analysis, its effectiveness measures are inferior to those of C-ICQ. In E-ICQ, samples are drawn from a coarser-grained uncertainty region derived based Euclidean distance. This leads to a less accurate contact distance and thus a lower probability of finding true close contact objects. The results show that our indoor uncertainty regions are more accurate for sampling locations.

Combining efficiency and effectiveness, C-ICQ works better than the others in close contact tracing.

Effect of δ. Figs. 9(a) and 9(b) report the efficiency of four methods, C-ICQ still costs least running time and memory because of our efficient model and strategies. The running time and memory cost of E-ICQ decrease when the distance constraint δ increases. This is because a higher δ makes it easier for the accumulating contact probability of object samples to exceed the fixed threshold η, causing a candidate object to be returned earlier and avoiding further processing in subsequent iterations. In contrast, C-ICQ and S-ICQ use indoor distances to find the uncertainty region, which includes fewer objects, resulting in relatively stable efficiency with increased δ. The time and memory costs of R-ICQ are insensitive to δ as there is no uncertainty analysis at unseen timestamps.

Figs. 9(c) and 9(d) report on the recall and F1 score. With a larger distance constraint, more objects tend to be counted as close contacts. As a result, the recall and F1 score measures of both methods increase. Compared to the two baselines, C-ICQ still performs best because of the precise analysis of the uncertainty region. R-ICQ ignores unseen sampling timestamps, so the variation of the distance constraint has no impact on its effectiveness.



(a) Time vs δ

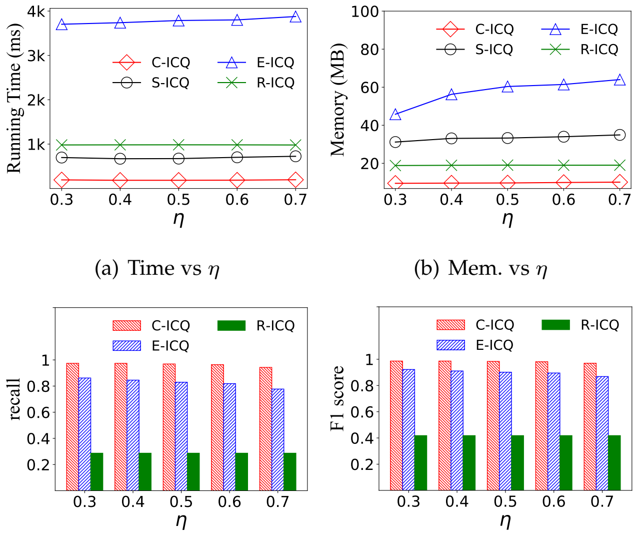
(b) Mem. vs δ

(c) Recall vs δ

(d) F1 score vs δ

Fig. 9: Effect of δ

Effect of η. Referring to Figs. 10(a) and 10(b), increasing the contact probability threshold η has little impact on the running time and memory consumption of all methods. Indeed, varying η does not change the number of calls of instant contact determinations.



(a) Time vs η

(b) Mem. vs η

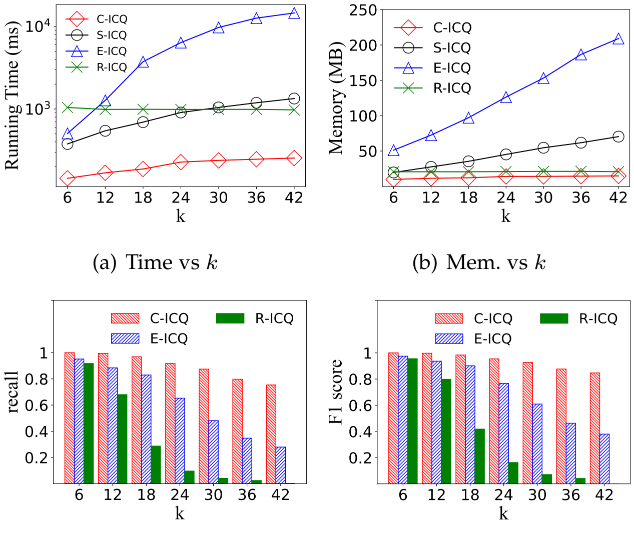
(c) Recall vs η

(d) F1 score vs η

Fig. 10: Effect of η

Figs. 10(c) and 10(d) report on the two effectiveness measures of the three methods when varying η. As mentioned above, R-ICQ omits unseen sampling timestamps. Therefore, both recall and F1 score of R-ICQ stay no change. In contrast, a higher η imposes stricter requirements on the establishment of instant contact, thereby making it less accurate the instant contact determination based on the derived samples. As a result, both recall and F1 score of C-ICQ and E-ICQ decrease with η increased. Nevertheless, C-ICQ still performs the best.

Effect of k. We vary k from 6 to 42 and report the running time and memory cost in Figs. 11(a) and 11(b), respectively. As no probabilistic samples are derived and processed, the running time and memory cost of R-ICQ are almost insensitive to k. In contrast, both running time and memory cost of other three methods increase with an increasing k. For E-ICQ, a larger k involves more sampling times to be checked consecutively in determining the close contact between two objects. This requires a longer running time and a larger memory cost. Compared to S-ICQ, C-ICQ's both measures increase much more slowly. This is due to its efficient time skipping strategy, which reduces the unnecessary heavy computations on consecutive sampling times. Moreover, the time and memory costs of C-ICQ tend to be stable when k is sufficiently large. This demonstrates that our proposed C-ICQ performs very well when the close contact tracing is conducted with a relatively large k.



(a) Time vs k

(b) Mem. vs k

(c) Recall vs k

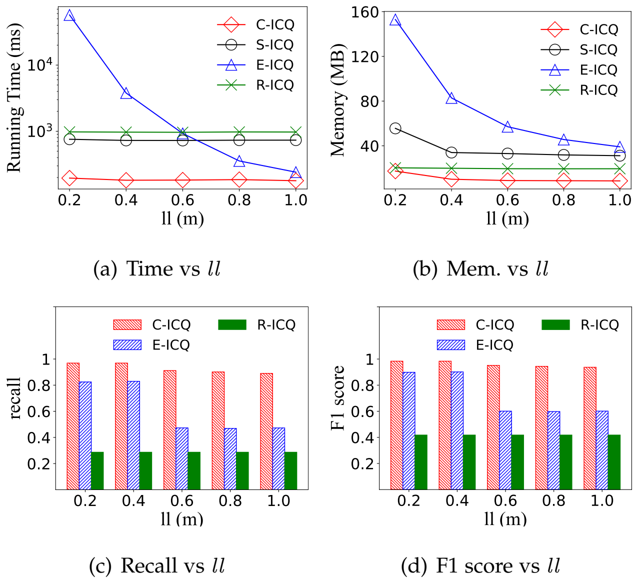
(d) F1 score vs k

Fig. 11: Effect of k

As shown in Figs. 11(c) and 11(d), the recall and F1 score of all methods deteriorate when k becomes larger. A larger k means a stricter requirement on close contact (see our research problem defined in Section 2.2), which renders the close contact tracing on uncertain positioning data less effective. Nevertheless, C-ICQ using probabilistic samples clearly outperforms E-ICQ analyzing uncertainty region with Euclidean distance and R-ICQ. Moreover, the performance gap enlarges when k increases to 42. In general, our proposed C-ICQ is more efficient and effective than the two baselines when a larger k is used.

Effect of ll. We vary the lattice side length ll from 0.2m to 1m. Referring to Figs. 12(a) and 12(b), in terms of the running time and memory cost, the efficiency of E-ICQ increases with a larger ll because fewer samples are introduced in computations. R-ICQ's time and memory costs remain stable as no analysis is involved. Still, C-ICQ runs much faster and cost less memory than other methods.

Referring to Figs. 12(c) and 12(d), the recall and F1 score of C-ICQ degrade when a larger ll is used. As fewer samples are used, the accuracy of instant contact determination



(a) Time vs ll

(b) Mem. vs ll

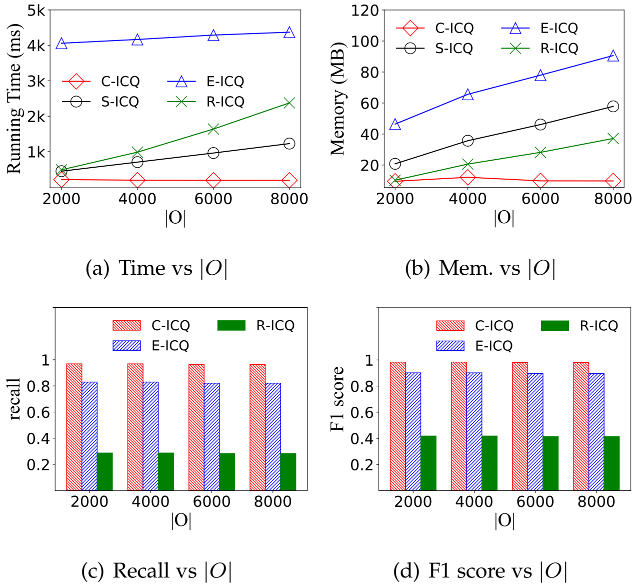
(c) Recall vs ll

(d) F1 score vs ll

Fig. 12: Effect of ll

decreases, and so do the recall and precision of the query result. In our experiments, C-ICQ achieves better recall and F1 score compared to the two baselines.

Effect of $|O|$. To test the scalability of all methods, we vary the object number $|O|$ from 2k to 8k. Referring Figs. 13(a) and 13(b), the time and memory costs of E-ICQ, R-ICQ and S-ICQ increases as $|O|$ increases, whereas C-ICQ stays stable. When more candidate objects are involved, E-ICQ, R-ICQ and S-ICQ need to check a larger fraction of potential close contact objects at each sampling time. In contrast, C-ICQ can rule out those unpromising objects and process only those necessary sampling times due to the time skipping strategy that it employs.



(a) Time vs $|O|$

(b) Mem. vs $|O|$

(c) Recall vs $|O|$

(d) F1 score vs $|O|$

Fig. 13: Effect of $|O|$

Referring to Figs. 13(c) and 13(d), both effectiveness measures of the three methods stay nearly stable when $|O|$ increases. Still, C-ICQ outperforms E-ICQ and R-ICQ due to its higher effectiveness of sample-based close contact tracing. Therefore, our proposed C-ICQ method works effectively in large-scale data scenarios.

5.3 Experiments on Real Data

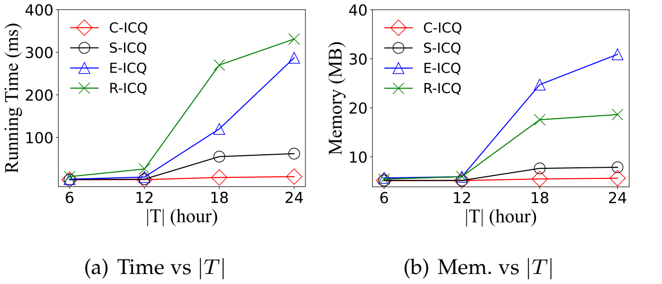
5.3.1 *Settings*

We also evaluate our two proposals on a real dataset collected from a shopping mall in Hangzhou, China. The shopping mall occupies 108m × 80m and 7 floors with 10 staircases. Each staircase is roughly 20m long. We acquire indoor positioning records from the building on 2018/01/01. In total, 4,616 distinct moving objects (MAC addresses) and 727,263 positioning records are obtained.

The query instance generation and the parameter settings follow the same as the experiments on the synthetic dataset (see Section 5.1). We do not vary $|O|$ as it is fixed in the real dataset. As we have no information about the true object movements to compute the recall and F1-score in the real data setting, we only compare the query processing methods in terms of efficiency metrics.

5.3.2 *Results*

Effect of $|T|$. We set the query time interval T to $(0, 6]$, $(0, 12]$, $(0, 18]$, and $(0, 24]$. The running time and memory costs are reported in Figs. 14(a) and 14(b), respectively. As the query interval length $|T|$ becomes larger, the running time and memory cost increase because more candidate objects and more sampling times are involved. Since fewer moving objects appeared in the morning, i.e., within (0, 12], but much more in the afternoon, i.e., within (12, 18], both costs of all methods increase significantly when T is changed from $(0, 12]$ to $(0, 18]$. In all tested cases, C-ICQ outperforms the three baselines in terms of efficiency.



(a) Time vs $|T|$

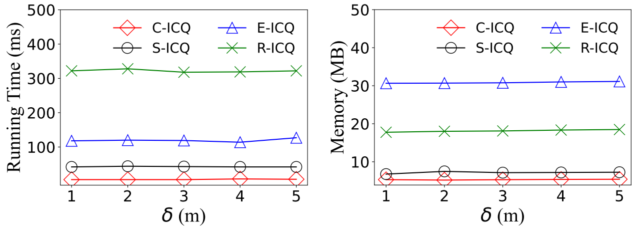
(b) Mem. vs $|T|$

Fig. 14: Effect of $|T|$

Effect of δ. According to the results reported in Figs. 15(a) and 15(b), both the running time and memory consumption stay stable for all methods as δ increases. Nevertheless, C-ICQ still performs best in terms of efficiency. By analyzing the distribution of the contact distances of the object pairs in the same partition, we find that there are just a few within a distance less than 5m.

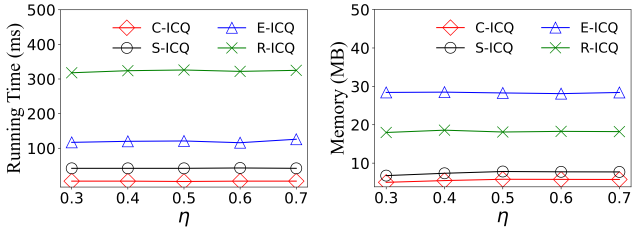
Effect of η. As shown in Figs. 16(a) and 16(b), the contact probability threshold η has no impact on the efficiency of R-ICQ because it omits the unseen sample timestamps. A larger η has a slight impact on C-ICQ since it has employed strategies for early termination of instant contact determination.

Effect of k. Figs. 17(a) and 17(b) report the running time and memory usage of all methods. Compared to C-ICQ, the baseline methods E-ICQ and R-ICQ cost more time and memory with an increasing k because our methods



(a) Time vs δ (b) Mem. vs δ

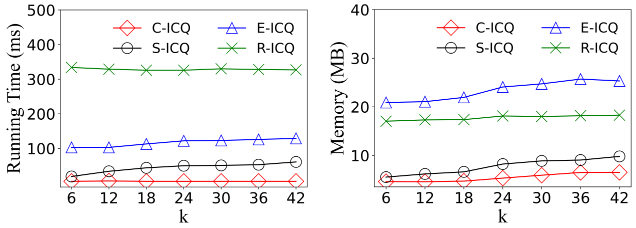
Fig. 15: Effect of δ



(a) Time vs η (b) Mem. vs η

Fig. 16: Effect of η

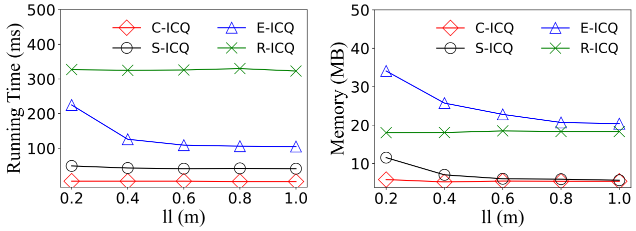
use the enhanced graph model that efficiently organizes the spatial data and trajectory and well supports the indoor contact tracing. The efficiency of both S-ICQ and C-ICQ decreases slightly when k increases. Increasing k involves more sampling times and thus increases both time and memory costs.



(a) Time vs k (b) Mem. vs k

Fig. 17: Effect of k

Effect of ll. According to the results reported in Figs. 18(a) and 18(b), the running time and memory consumption decrease as ll increases. In general, increasing ll has no impact on R-ICQ and only slight impact on C-ICQ, but improves significantly the efficiency of E-ICQ. These observations are consistent with what we have seen on the synthetic data.



(a) Time vs ll (b) Mem. vs ll

Fig. 18: Effect of ll

5.4 Summary of Experimental Results

We summarize our findings from the results as follows.

In terms of efficiency, C-ICQ performs best because it avoids a lot of heavy computations with efficient acceleration strategies, while S-ICQ needs more time and memory costs because it analyzes the close contacts for all relevant sampling times without any acceleration.

On the other hand, S-ICQ and C-ICQ return the same results. Their query result effectiveness is always better than that of E-ICQ and R-ICQ. This demonstrates the necessity of our proposed sample-based close contact tracing, which captures indoor moving objects' spatiotemporal dynamics much better than the E-ICQ using Euclidean distance and the R-ICQ ignoring the unseen sampling timestamps.

All in all, considering both efficiency and effectiveness, C-ICQ is the best among all methods in comparison.

6 Related Work

Contact Tracing. There are two mainstream contact tracing technologies, namely proximity-based solutions [3], [4], [5], [6], [7], [8], [9], [25], [26], [27], [28], [29] and location-based solutions [10], [11], [12], [13], [14], [15], [16], [17], [21].

Relying on wireless technologies such as Bluetooth, the proximity-based contact tracing solutions regard two devices as close contacts of each other if one of them falls within the wireless detection range of the other for a certain time period. Some studies review and compare proximity-based contact tracing methods, from the perspectives of system design [27], [28], privacy issues [29], and future development [25], [26]. Some recent representative works are introduced below. Tedeschi et al. [3] propose an IoT-based architecture for saving energy consumption and computational cost on end-user devices. Ng et al. [5] predict the exposure risk of a user using a machine learning classifier. Shrestha et al. [8] propose a service-oriented architecture to enable Bluetooth-based mobile contact tracing. Li et al. [7] study privacy-aware, indirected close contact using the hashed WiFi access point IDs in a distributed environment. The proximity-based methods, however, suffer from several shortcomings. First, these methods require mobile app installation, which is less user-friendly. Second, they fail to support adjusting criteria of close contacts, e.g., customizing the distance restriction. Third, they neglect spatial information, not to mention indoor topology.

Location-based contact tracing methods identify close contact pairs using historical location data instead of wireless proximity information. Different from our work on efficient contact tracing over historical data, many works focus on developing the contact tracer as an administrative web-based system [11], a privacy-enhanced mobile application [10], [12], [15], or a decentralized Blockchain-based system [16]. Several works [13], [14], [17] consider the efficiency of close contact search. In particular, Chao et al. [14] devise an iterative algorithm and a grid-based time interval tree to find contacted trajectories along the transmission chains. Li et al. [17] develop group processing algorithms for online continuous monitoring of the contacts with the given objects. Zhang et al. [13] resolve contact similarity search on uncertain trajectories based on an uncertain trajectory M-tree applied to necklace-based trajectories. However, all these aforementioned studies apply to GPS data and determine the contact based on free-space distances. As a result, they cannot resolve our problem in indoor settings characterized by unique indoor topology and distances. Some works focus on indoor contact tracing. Alarabi et al. [30] adapt a three-dimensional R-Tree to answer the contact tracing queries. However, they do not consider the indoor restrictions such as walls and doors. Ho et al. [31] develop a system to analyze the movement of users, as a foundation to further help contact tracing, but it can not solve contact tracing directly. Moreover, neither of the works [30], [31] considers the uncertainty of indoor data.

Analysis and Queries over Indoor Uncertain Data. Lu et al. [18] propose and study probabilistic, spatio-temporal joins on historical RFID-like tracking data, which aim to find the object pairs in the same room at a particular timestamp. However, such joins do not consider concrete indoor distances. Xie et al. [32] study indoor distance-aware join on moving objects in indoor space. However, the join query returns indoor object pairs at a certain time and it does not consider the duration. Lu et al. [33] mine frequently visited POIs from symbolic indoor tracking data collected through Bluetooth technologies. The data formats of these works [18], [32], [33] are different from that in ICQ. Li et al. [19] propose a framework for finding the current top-k indoor dense regions from a set of user-defined query regions. Li et al. [20] tackle the problem of finding the top-k popular indoor semantic locations with the highest flow values using uncertain historical indoor mobility data. Nonetheless, these works [19], [20], [33] return dense or popular indoor regions, while our work concerns the relationship between two individual objects. Liu et al. [34] study crowd-aware indoor path planning queries by taking moving objects into account in an online setting. Unlike ICQ, this work [34] does not analyze the uncertain regions of individual objects. A recent work [21] considers indoor topology and uncertainty modeling in continuously monitoring the distances between object pairs. However, the proposed solution aims for early warning of all approaching object pairs by continuously calculating and forecasting their indoor distances. In contrast, our study focuses on retrieving from historical data past close contacts with a confirmed infected object.

7 Conclusion and Future Work

In this work, we study the novel Indoor Contact Query (ICQ). It conducts contact tracing over uncertain indoor positioning data to find all moving objects that had close contact with a given object o. To process ICQ, we propose a set of data management techniques: 1) an enhanced indoor graph model to organize a multitude of relevant data; 2) an uncertainty-aware method for determining instant close contact; 3) a unified query processing framework with a close contact determination method, a search algorithm, and acceleration strategies. Experiments on synthetic and real datasets demonstrate the efficiency and effectiveness of our proposals.

Several directions exist for future research. First, it is interesting to support other criteria of close contact within our proposed framework. Also, it is possible to generalize our solution to support contact tracing over different types of indoor positioning data. Moreover, it is relevant to consider contact tracing over a mix of indoor and outdoor mobility data when such data is available. Last but not least, in case that COVID-19 rebounds, it is interesting to construct and deploy a prototype of an ICQ system in a real-world setting, such as a university campus, to help contain virus spread.

Acknowledgement

This work was supported by Independent Research Fund Denmark (No. 8022-00366B) and Australian Research Council (DP230100081 and FT180100140).

References

[1] https://knowablemagazine.org/article/health-disease/2020/pandemics-recent-history.

[2] A. Baniukevic, D. Sabonis, C. S. Jensen, and H. Lu, "Improving wi-fi based indoor positioning using bluetooth add-ons," in *MDM*, vol. 1. IEEE, 2011, pp. 246–255.

[3] P. Tedeschi, S. Bakiras, and R. Di Pietro, "Iotrace: A flexible, efficient, and privacy-preserving iot-enabled architecture for contact tracing," *IEEE Communications Magazine*, vol. 59, no. 6, pp. 82–88, 2021.

[4] R. L. Rivest, D. Weitzner, L. Ivers, I. Soibelman, and M. Zissman, "Pact: Private automated contact tracing," *Retrieved December*, vol. 2, p. 2020, 2020.

[5] P. C. Ng, P. Spachos, S. Gregori, and K. N. Plataniotis, "Epidemic exposure tracking with wearables: A machine learning approach to contact tracing," *IEEE Access*, 2022.

[6] J. Bay, J. Kek, A. Tan, C. S. Hau, L. Yongquan, J. Tan, and T. A. Quy, "Bluetrace: A privacy-preserving protocol for community-driven contact tracing across borders," *GovTech*, vol. 18, 2020.

[7] G. Li, S. Hu, S. Zhong, W. L. Tsui, and S.-H. G. Chan, "vContact: Private WiFi-based IoT contact tracing with virus lifespan," *IEEE IoT-J*, vol. 9, no. 5, pp. 3465–3480, 2021.

[8] M. Shrestha, X. Liu, and M. Sreekanthan, "A Bluetooth-based contact-tracing mobile app for airborne-based epidemic control," in *ISNCC*. IEEE, 2021, pp. 1–5.

[9] P. Di Marco, P. Park, M. Pratesi, and F. Santucci, "A bluetooth-based architecture for contact tracing in healthcare facilities," *JSAN*, vol. 10, no. 1, p. 2, 2020.

[10] L. Xiong, C. Shahabi, Y. Da, R. Ahuja, V. Hertzberg, L. Waller, X. Jiang, and A. Franklin, "REACT: Real-time contact tracing and risk monitoring using privacy-enhanced mobile tracking," *SIGSPATIAL Special*, vol. 12, no. 2, pp. 3–14, 2020.

[11] S. S. Eusuf, K. A. Islam, M. E. Ali, S. M. Abdullah, and A. S. Azad, "A Web-based system for efficient contact tracing query in a large spatio-temporal database," in *SIGSPATIAL*, 2020, pp. 473–476.

[12] F. Kato, Y. Cao, and M. Yoshikawa, "PCT-TEE: Trajectory-based private contact tracing system with trusted execution environment," *TSAS*, vol. 8, no. 2, pp. 1–35, 2021.

[13] X. Zhang, S. Ray, F. Shoeleh, and R. Lu, "Efficient contact similarity query over uncertain trajectories," in *EDBT*, 2021, pp. 403–408.

[14] P. Chao, D. He, L. Li, M. Zhang, and X. Zhou, "Efficient trajectory contact query processing," in *DASFAA*. Springer, 2021, pp. 658–666.

[15] J. Ami, K. Ishii, Y. Sekimoto, H. Masui, I. Ohmukai, Y. Yamamoto, and T. Okumura, "Computation of infection risk via confidential locational entries: A precedent approach for contact tracing with privacy protection," *IEEE Access*, 2021.

[16] H. R. Hasan, K. Salah, R. Jayaraman, I. Yaqoob, M. Omar, and S. Ellahham, "COVID-19 contact tracing using Blockchain," *IEEE Access*, vol. 9, pp. 62 956–62 971, 2021.

[17] K. Li, L. Chen, S. Shang, Y. Liu, H. Wang, P. Kalnis, and B. Yao, "Towards controlling the transmission of diseases: Continuous exposure discovery over massive-scale moving objects," in *IJCAI*. ijcai. org, 2022.

[18] H. Lu, B. Yang, and C. S. Jensen, "Spatio-temporal joins on symbolic indoor tracking data," in *ICDE*. IEEE, 2011, pp. 816–827.

[19] H. Li, H. Lu, L. Shou, G. Chen, and K. Chen, "In search of indoor dense regions: An approach using indoor positioning data," *IEEE TKDE*, vol. 30, no. 8, pp. 1481–1495, 2018.

[20] ——, “Finding most popular indoor semantic locations using uncertain mobility data,” *IEEE TKDE*, vol. 31, no. 11, pp. 2108–2123, 2018.

[21] H. K.-H. Chan, H. Li, X. Li, and H. Lu, “Continuous social distance monitoring in indoor space,” *Proc. VLDB Endow.*, vol. 15, no. 7, pp. 1390–1402, 2022.

[22] H. Lu, X. Cao, and C. S. Jensen, “A foundation for efficient indoor distance-aware query processing,” in *ICDE*. IEEE, 2012, pp. 438–449.

[23] M. Boysen, C. de Haas, H. Lu, and X. Xie, “A journey from ifc files to indoor navigation,” in *W2GIS*. Springer, 2014, pp. 148–165.

[24] M. Boysen, C. de Haas, H. Lu, X. Xie, and A. Pilvinyte, “Constructing indoor navigation systems from digital building information,” in *ICDE*. IEEE, 2014, pp. 1194–1197.

[25] T. D. Nguyen, M. Miettinen, A. Dmitrienko, A.-R. Sadeghi, and I. Visconti, “Digital contact tracing solutions: Promises, pitfalls and challenges,” *arXiv preprint arXiv:2202.06698*, 2022.

[26] A. Blasimme, A. Ferretti, and E. Vayena, “Digital contact tracing against COVID-19 in europe: Current features and ongoing developments,” *Frontiers in Digital Health*, vol. 3, p. 61, 2021.

[27] L. Reichert, S. Brack, and B. Scheuermann, “A survey of automatic contact tracing approaches using Bluetooth Low Energy,” *ACM HEALTH*, vol. 2, no. 2, pp. 1–33, 2021.

[28] N. Ahmed, R. A. Michelin, W. Xue, S. Ruj, R. Malaney, S. S. Kanhere, A. Seneviratne, W. Hu, H. Janicke, and S. K. Jha, “A survey of COVID-19 contact tracing apps,” *IEEE Access*, vol. 8, pp. 134 577–134 601, 2020.

[29] M. Hatamian, S. Wairimu, N. Momen, and L. Fritsch, “A privacy and security analysis of early-deployed covid-19 contact tracing Android apps,” *Empirical Software Engineering*, vol. 26, no. 3, pp. 1–51, 2021.

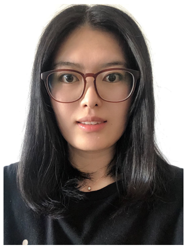
[30] L. Alarabi, S. Basalamah, A. Hendawi, and M. Abdalla, “Traceall: A real-time processing for contact tracing using indoor trajectories,” *Information*, vol. 12, no. 5, p. 202, 2021.

[31] Y. A. Ho, C. K. Tan, and Y. H. Ng, “Clustering indoor location data for social distancing and human mobility to combat covid-19,” *Computers, Materials and Continua*, vol. 71, no. 1, pp. 907–924, 2022.

[32] X. Xie, H. Lu, and T. B. Pedersen, “Distance-aware join for indoor moving objects,” *IEEE TKDE*, vol. 27, no. 2, pp. 428–442, 2014.

[33] H. Lu, C. Guo, B. Yang, and C. S. Jensen, “Finding frequently visited indoor pois using symbolic indoor tracking data.” in *EDBT*, 2016, pp. 449–460.

[34] T. Liu, H. Li, H. Lu, M. A. Cheema, and L. Shou, “Towards crowd-aware indoor path planning.” *Proc. VLDB Endow.*, vol. 14, no. 8, pp. 1365–1377, 2021.

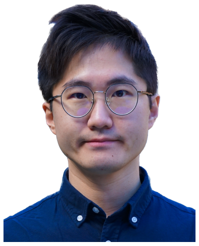


Tiantian Liu is a Postdoc with the Department of People and Technology, Roskilde University, Denmark. She received the BSc and MSc degrees from Jilin University, China, and the PhD degree in the Department of Computer Science, Aalborg University, Denmark. Her research interests include spatio-temporal data analytics, graph search, and data management.

Huan Li is a ZJU 100 Young Professor at Zhejiang University and a recipient of an EU Marie Curie Individual Fellowship. He was an Assistant Professor at Aalborg University in Denmark from 2020 to 2022 and a Senior Engineer at Alibaba from 2018 to 2019. He received his PhD from Zhejiang University in 2018. His research focuses on IoT data management, spatiotemporal AI, and mobile computing, and most of his works have been published in top-tier journals and conferences. He is a member of IEEE.

Hua Lu is a Professor of Computer Science in the Department of People and Technology, Roskilde University, Denmark. He received the BSc and MSc degrees from Peking University, China, and the PhD degree in computer science from National University of Singapore. His research interests include data management, spatial data, location-based services, data science and GIS. He has served as PC cochair or vice chair for ISA 2011, MUE 2011, MDM 2012, NDBC 2019 and IEEE BigData 2022, demo chair for SSDBM 2014, and PhD forum cochair for MDM 2016 and MDM 2022. He has served on the program committees for conferences such as VLDB, ICDE, KDD, WWW, CIKM, DASFAA, ACM SIGSPATIAL, SSTD and MDM. He received the Best Vision Paper Award at SSTD 2019. He is a senior member of the IEEE.

Muhammad Aamir Cheema is an ARC Future Fellow and Associate Professor at the Faculty of Information Technology, Monash University, Australia. He obtained his PhD from UNSW Australia in 2011. He is the recipient of 2012 Malcolm Chaikin Prize for Research Excellence in Engineering, 2013 Discovery Early Career Researcher Award, 2014 Dean’s Award for Excellence in Research by an Early Career Researcher, 2018 Future Fellowship, 2018 Monash Student Association Teaching Award and 2019 Young Tall Poppy Science Award. He has also won two CiSRA best research paper of the year awards, two invited papers in the special issue of IEEE TKDE on the best papers of ICDE, and three best paper awards at ICAPS 2020, WISE 2013 and ADC 2010, respectively. He is the Associate Editor of IEEE TKDE and DAPD and served as PC co-chair for ADC 2015, ADC 2016, 8th ACM SIGSPATIAL Workshop ISA 2016 & 2018, IWSC 2017, proceedings chair for DASFAA 2015 & ICDE 2019.



Harry Kai-Ho Chan is a Lecturer in Data Science at Information School, University of Sheffield, United Kingdom. He received the BEng, MPhil and PhD degrees in computer science and engineering from the Hong Kong University of Science and Technology. His research interests include spatio-temporal data management, data mining, and data science. He has served on the program committees for conferences such as KDD, ICDE, CIKM, SSTD and IEEE BigData. He is a member of IEEE.